

# QC Results Overview

QC Results is a multi-tier data collection system used on both the shop-floor and in the back-office. Paper QC forms used in the past can be replaced with new electronic forms. The QC Results forms write to basic SQL Server database tables which can store things such as numbers, pictures, notes and time/date information. From there reporting systems can be built to closely track and analyze the data coming from your plant or warehouse.

This document will cover the various controls and features found inside the Coolearth QC Results system.

## Platforms

There are currently two UI platforms that QC Results runs on.

WPF (Windows Presentation Foundation): This is the desktop platform; the QC Results client is launched and runs from inside Whistle Studio. It can run on machines with Windows XP/2003 or greater and the .Net Framework 3.5 SP1 installed. Most of the screenshots contained within this document are from the WPF client.

Windows Mobile 6.1: This is the handheld platform; any handheld device running Windows Mobile 6.1 (or greater) and the .Net Compact Framework 3.5 can run the handheld client.

## Coolearth Whistle Studio

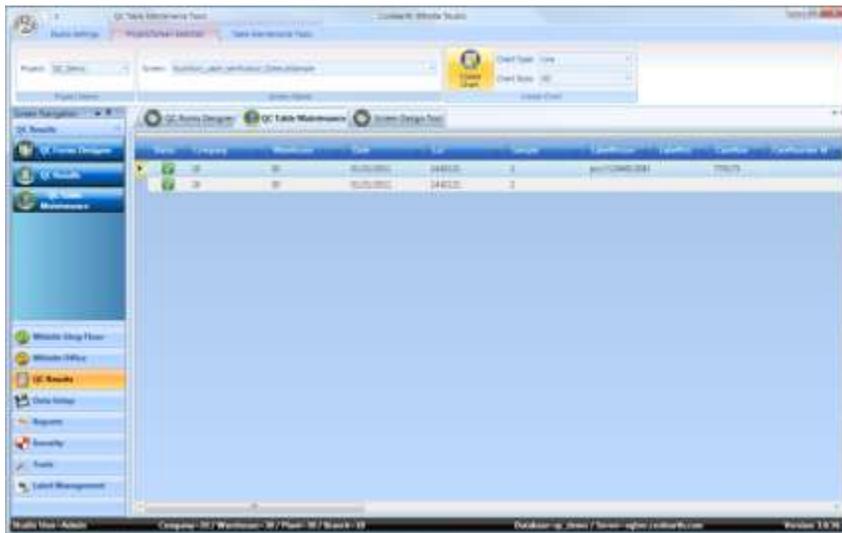


Figure 1 - Studio with QC Table Maintenance screen active.

Many of the QC Results back-office functions are performed inside Whistle Studio. The WPF client is run inside the Studio. Studio screens are launched by clicking the buttons on the left that are organized in a Microsoft Outlook style navigation pane.

## Logging In



Figure 2 - Studio screen button for the QC Results WPF client.

Logging into the QC Results is done using either a user name and password set up in the ERP system or if running standalone a user defined SQL Server database table. Inside the desktop version, the credentials used for Studio login are automatically passed down to the QC Results system. In the handheld client, the credentials are entered at a login screen.

## Projects, Screens, and Controls

QC Results is made up of one or more structures called projects. Each project is made up of a collection of screens which are in turn made up of a collection of controls. Each screen gets its own backing database table that contains the data collected using the various controls. Most of the controls have one or more columns in the backing table that are tied to the control's value. So when a user edits a value inside a text box for instance, the QC Results application persists that value into the backing table. Usually projects are broken up on a warehouse by warehouse basis, that is, all the screens contained in a project are designed around replacing the paper forms for a given warehouse. However, the screen designer can choose to break up however they would like.

## Menus

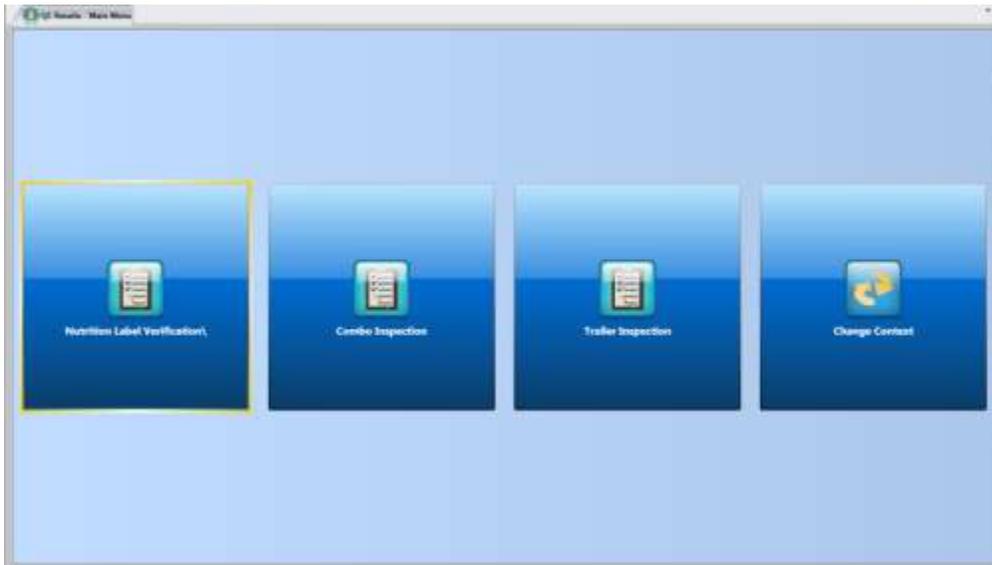


Figure 3 - Main menu for the QC\_Demo project.

The menu system in QC Results is hierarchical. The user starts out on what is known as the root menu (or Main Menu) and descends into the various sub menus until a set of forms is reached. In that sense,

groups of screens can be logically organized to suit the QC department's needs. In our simplistic test project we have no sub-menus, just three forms that are accessible from the root menu.

The 'Change Context' screen pictured above is used for switching which company and warehouse the user is working inside. The company and warehouse of the user's context are written as metadata fields inside each row of the backing table.

## Root Screens

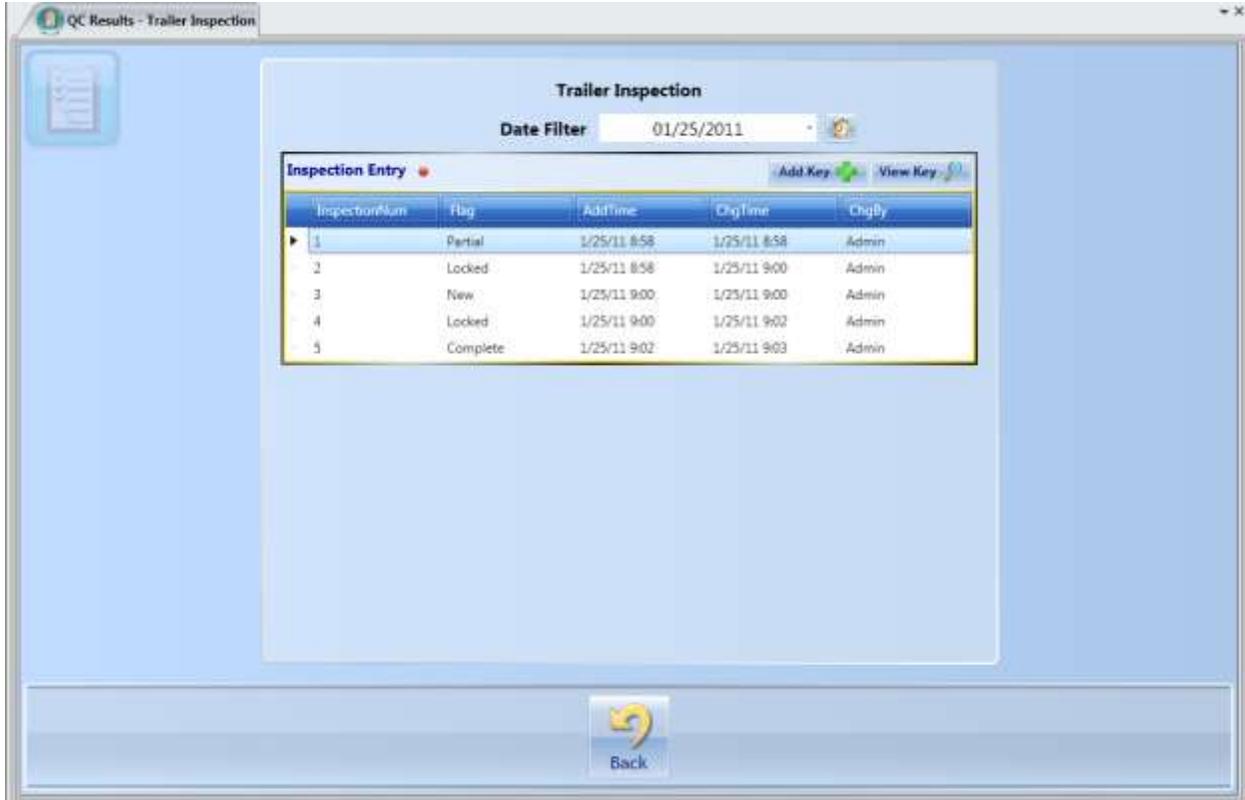


Figure 4 - Trailer Inspection Root Screen.

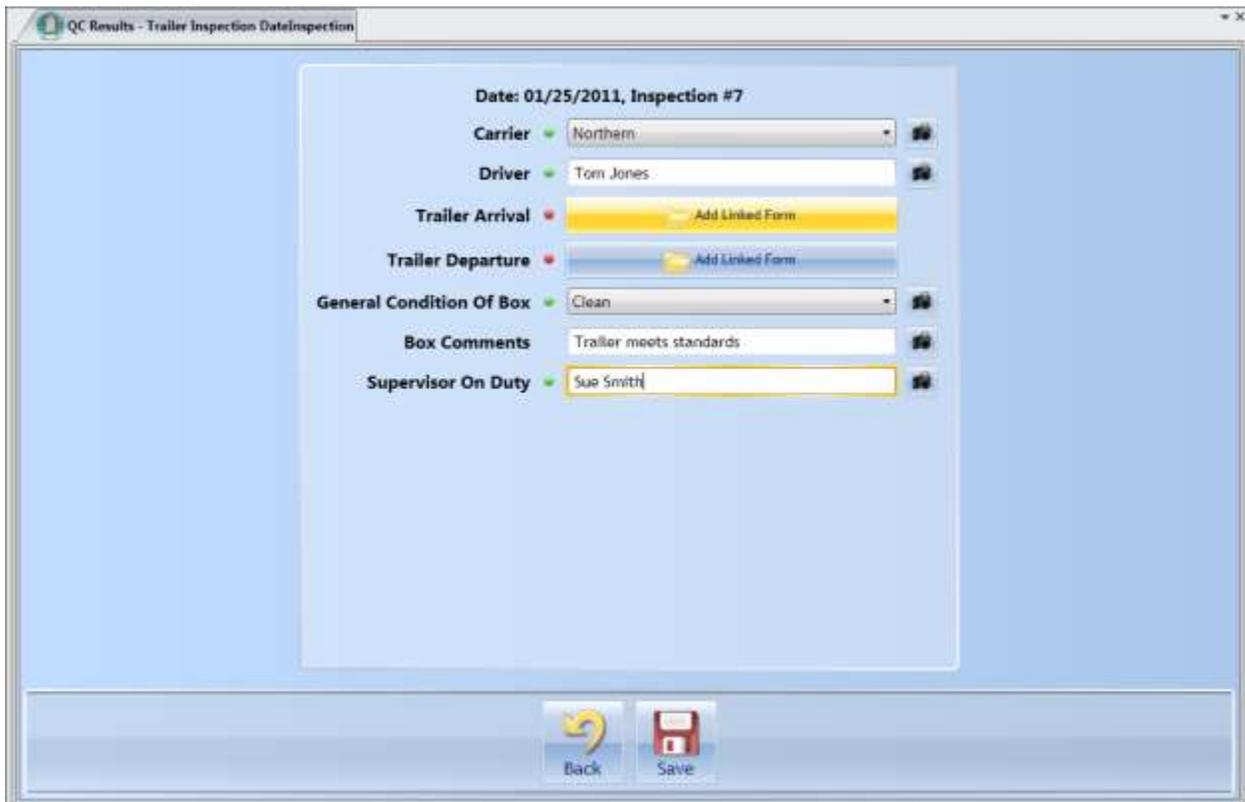
This screenshot is from the root screen of the *Trailer Inspection* form. Most often root screens simply serve as an entry point into the lower screens that actually collect data. They are an exception in that the root screens most often do not contain a backing database table. Whether or not a screen contains a backing table is determined by if the screen has any keys. If the screen has no keys, no backing table is used. In this screen there are three fields, a label with the basic screen title, a Date Editor (labeled Date Filter) for filtering and creating new entries, and a Drill-Down Grid (labeled Inspection Entry) for adding and viewing child rows. For an explanation of controls and how they work, see the following section – *QC Results Controls*.

## CoolScript

One of the most powerful tools available to the screen designer is the ability to embed scripting code inside the controls/screen events. Inside the screen pictured above, there are several pieces of CoolScript working in the background. For instance inside the Date Editor, there is scripting code wired up to the OnFieldChanged event that causes the Shift Entry grid to reload its contents based on the selected date. Inside the Shift Entry Drill-Down Grid control, there is scripting code to navigate to the next child screen that fires off when the 'Add Key' and 'View Key' buttons are pressed.

The CoolScript can be written to pass down any number of different arguments that might be of interest to the child screen. Most often the script inside the Drill-Down Grid control involves refreshing the content of the grid when a new row is added, or passing the entered keys down to the next child screen (in this case date and inspection number are passed down as arguments).

## Child Screens



The screenshot shows a software window titled "QC Results - Trailer Inspection DateInspection". The main content area contains a form with the following elements:

- Date:** 01/25/2011, Inspection #7
- Carrier:** Northern (dropdown menu)
- Driver:** Tom Jones (text input)
- Trailer Arrival:** Add Linked Form (button)
- Trailer Departure:** Add Linked Form (button)
- General Condition Of Box:** Clean (dropdown menu)
- Box Comments:** Trailer meets standards (text input)
- Supervisor On Duty:** Sue Smith (text input)

At the bottom of the window, there are two buttons: "Back" (with a yellow arrow icon) and "Save" (with a red floppy disk icon).

Figure 5 - Trailer Inspection Child Screen.

Above is a child screen that has been broken up into two different sections with the Form Link Control. Notice that this screen has a 'Save' button that becomes enabled when there is unsaved data that can be persisted to the backing table. Also note the title of this screen - *Trailer Inspection DateInspection*. Often when naming child screens it is a good practice to indicate the keys that the screens are based around, so in this case Date and Inspection are the keys. The name of the backing database table is

derived from the project and screen name. For example, the *Trailer Inspection DateInspection* screen's backing database table is named `qcQCDEMO_TrailerInspectionDateInspection`. All QC Results backing tables begin with the letters 'qc.'

By clicking one of the Form Link Controls (Trailer Arrival or Trailer Departure), the user is navigated to another child screen for filling out the trailer arrival and departure sections respectively. Also, you will notice two common features here that are found in almost all QC Results screens:

## Mandatory Indicator / Filled In

Each of the fields inside QC Results can be marked as being mandatory or non-mandatory. Fields marked as being mandatory will have a red or green indicator that appears to the left of them. A field that is marked as mandatory must be filled in before the screen is considered 'Complete'. Only a screen that has a status of 'Complete' can be put into the 'Locked' state (see the next section—Screen States).



- Mandatory controls that have been filled in will have a green indicator.



- Mandatory controls that are not filled in show a red indicator.

What determines if a control considers itself filled in or not depends on the type of control. For instance, a Text Editor is considered filled in if the user has entered text into the control. A Pass-Fail Box is considered filled in if the user has clicked on either pass or fail. Alternatively, there is an option that can be turned on which causes a Pass-Fail Box to not consider itself filled unless there is a comment entered after choosing a pass/fail value. The Form Link Control doesn't consider itself filled in until the child forms that live underneath it have a status of 'Locked' (see above screenshot). Similarly, the Drill-Down Grid control doesn't become filled in until it has one or more child rows in the 'Locked' state. The number of locked rows a Drill-Down grid needs to consider as filled in is configurable through a control option.

## Screen States

The child screens inside QC Results each correspond to a row inside the backing database table. Every row in the backing database table has a common set of fields called the Row Metadata. The time the row was created and who created it, the last time the row was modified and the user who modified it, and the status of the row are all part of the Row Metadata. The list of screen states (also known as row status) is as follows:

1. "New" – The row has been created inside the database but has no data beyond the basic Row Metadata filled out.
2. "Partial" – The row has been created and some of the fields have been filled out but not all mandatory controls have been filled in.
3. "Complete" – All the mandatory fields for the record have been filled in, but the user has not signed off and locked the form by clicking the 'Done' button.

4. “Locked” – The user has filled in all mandatory fields and signed off on the form by clicking the ‘Done’ button. Once a form has been signed off on, it becomes read-only and can be viewed but not modified from that point on.

## Attach Image Metadata



- Indicates the field supports image attachments, but no images have been uploaded.



- One or more images have been uploaded and are attached to the field.

One of the most powerful features in QC Results is the ability to attach images to a field and store them inside a table as part of the Field Metadata. Images are collected on the shop floor using the handheld’s built in camera and sent wirelessly to the server where they are persisted into the database.

Alternatively, images (possibly taken with a digital camera) can be uploaded from the desktop environment by simply selecting a file located on the user’s hard drive by using the image metadata editor control. Controls that support image metadata appear with an image metadata indicator beside them (pictured above). Clicking on the image metadata indicator button will launch an image metadata editor control (pictured below).



Figure 6 - Attach image metadata control.

With the image metadata editor control, users can add, delete and navigate through images attached to a given field on a form.

## Constraints

Another powerful feature of the QC Results system is the ability to set up constraints on a field. A constraint is defined as a value or range of values which the data entered must fall inside. There are four types of constraints that can be defined for numeric fields:

1. Hard Maximum – This is the maximum tolerable value that the field can have. Anything greater than the hard maximum will cause the field to turn red, alerting the user that the reading entered is out of spec and required corrective action.
2. Soft Maximum – Any readings greater than the soft maximum cause the field to turn yellow, indicating that the value has exceeded a soft threshold but is still less than the hard maximum.
3. Soft Minimum – Works the same as soft maximum but in the reverse direction. If the value entered is less than the soft minimum then the field will turn yellow.
4. Hard Minimum – Same as hard maximum except reversed.



The screenshot shows a web application window titled "QC Results - TrailerInspection\_DateInspection\_Arrival". The main form is titled "Trailer Inspection - Arrival" and contains several fields: "Arrival Time" (09:56 AM), "Refrigerator Operating" (Yes), "If No" (empty), "Temp At Arrival" (-8.00), "Adjusted Arrival Temperature" (empty), and "Arrival Comments" (empty). The "Temp At Arrival" field is highlighted in yellow, indicating a soft maximum constraint failure. A modal dialog box titled "Arrival Temperature" is displayed over the form, containing the message: "Trailer arrival temperature too high (> -10 degrees), adjust temperature and record below." and an "OK" button. At the bottom of the window, there are "Back" and "Save" buttons.

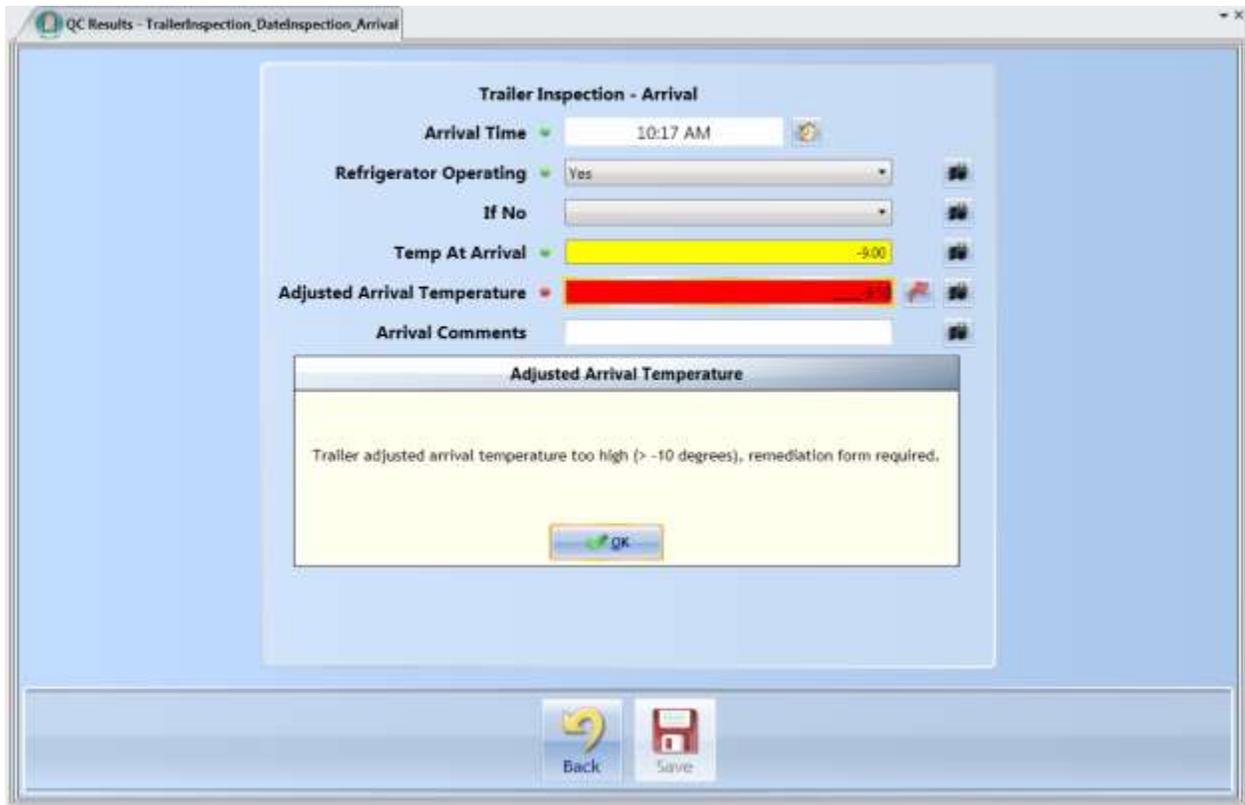
Figure 7 - Temp At Arrival field with a soft maximum constraint failure.

For each of the four types of constraint failures, a CoolScript event is triggered. The screen designer may choose to embed scripting code which launches a corrective action. Examples of corrective action

might be firing an e-mail off to supervisors, requiring the user to fill in additional fields describing what they did to address the problem, or launching a full blown corrective action report form.

In the screenshot above, the user has entered a value of -9.0 degrees which has exceeded the soft maximum constraint. The embedded CoolScript code displays a message box and sets the Adjusted Arrival Temp field as a mandatory field, forcing the inspector to adjust the trailer temperature and record the adjusted value.

## Linked Remediation Forms



The screenshot displays a web application window titled "QC Results - TrailerInspection\_DateInspection\_Arrival". The main content area is titled "Trailer Inspection - Arrival" and contains several input fields: "Arrival Time" (10:17 AM), "Refrigerator Operating" (Yes), "If No" (empty), "Temp At Arrival" (-9.00), "Adjusted Arrival Temperature" (-9.5), and "Arrival Comments" (empty). A modal dialog box titled "Adjusted Arrival Temperature" is overlaid on the form, containing the message: "Trailer adjusted arrival temperature too high (> -10 degrees), remediation form required." and an "OK" button. At the bottom of the application window, there are "Back" and "Save" buttons.

Figure 8 - Trailer Inspection Arrival screen with Linked Remediation Form.

When a constraint has failed, the scripting code can launch a linked remediation form. Take our above example where the trailer arrival temperature is too high. Now suppose the inspector is unable to get the adjusted temperature under the threshold of -10.0 degrees. So the inspector enters in -9.5 as the adjusted temperature and a message box appears informing them that they must fill out a remediation form. After the inspector clicks OK, they are taken to a Linked Remediation Form.

QC Results - TrailerInspection\_Remediation

**Trailer Inspection Remediation Form**

Remediation For: Adjusted Arrival Temperature

Temperature: -9.50

Description  Could not get trailer temperature under the required value of -10.0 degrees. 

Action Taken  Refused Shipping 

Adjusted Temperature   

Notified Supervisor  Yes 

Supervisor Name  Harrison Jones 

Comments  Refused shipping, trailer temperature is too warm. 

Back Save Done

Figure 9 - Remediation form for the Trailer Inspection screen.

The status of the linked remediation form is indicated by the arrow icon:



- A red arrow means the linked remediation form is in the 'Created' state.



- A yellow arrow means the linked remediation form is in the 'Partial' state.



- A green arrow indicates that the linked remediation form is in the 'Complete' state but hasn't been locked.



- A green arrow with a padlock on it indicates the linked remediation form has been filled out and in the 'Locked' state.

Once a field's remediation form is in the 'Locked' state, the field will then be considered filled-in and the rest of the form may be completed.

## Using CoolScript to Constrain User Input and Fetch Data



Figure 10 - Lot Entry Drill-Down with constrained user input.

The CoolScript scripting language is used just about everywhere when building screens in QC Results. Pictured above is a screen *Nutrition Label Verification* that only allows the user to select from an ordered list of valid lots. The CoolScript behind the `OnAddNewKeyItem` event handler does a SQL query against the ERP system's lot table and fills the Lot Entry drop-down with the results.



Figure 11 - Nutrition Label Verification screen with case number 770175 entered.

In the above example, we have entered case number 770175 into the Case Number field. The CoolScript behind the OnFieldChanged event handler for this control executes a SQL query that checks the WMS system's case table to determine if the entered case number really does belong to lot 1410121. If it does, the item description and label weight are pulled from the database table and displayed on the screen. The label weight that was looked up from the database is also used further down in the screen when the inspector enters the QA Scale Weight field. The CoolScript behind this field calculates the difference between the weight in the database and the entered scale weight and displays the difference. The weight difference is then stored in what is known as a dialog variable.

## Dialog Variables

A dialog variable is a CoolScript entity which stores its value in the backing SQL Server database table but isn't part of a control that shows up on the screen. It acts just like a regular programming variable, and can take on many different forms such as an integer, decimal, date-time, or Boolean value. What makes a dialog variable interesting is that its value is persisted and de-persisted along with the screen's contents. In the above example pictured in figure 11, there is a dialog variable named WeightDifference that holds and stores the calculated weight difference.

## QC Results Table Maintenance



So far this document has only gone over the features of the QC Results client application. Now we will briefly go over the backing SQL Server database tables.

The QC Results system comes with a screen that resides inside of Studio which allows users to view and maintain the data that has been collected. Typically, access to this screen would only be available to administrators. There are basically two different modes to this screen that are governed by security permissions, one that lets you view the data and one that lets you both view and edit the data. Normally you wouldn't want to change the data that has been collected, but it is possible in this system for super-administrators to have this ability should they ever need it.

Below is a screenshot that shows a row being edited that we created earlier with the QC Results client.

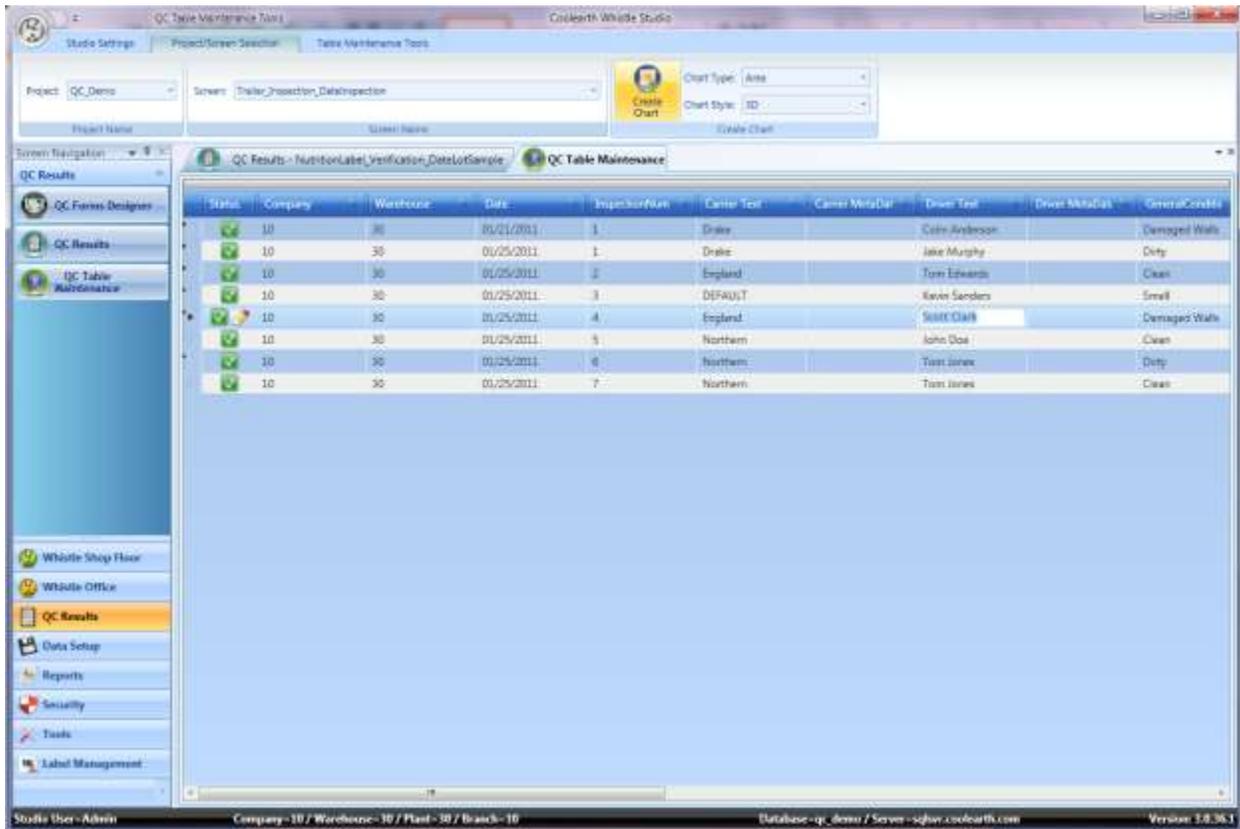
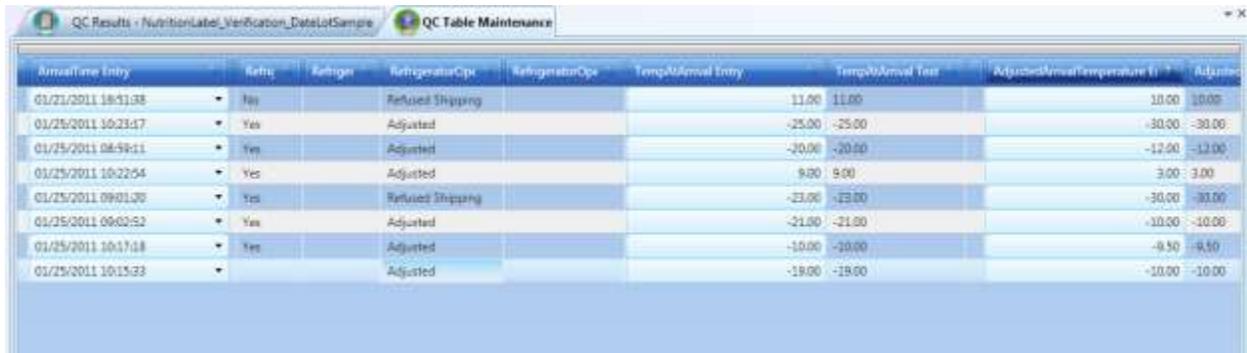


Figure 12 - User editing a field with the QC Table Maintenance.

A list of available projects and screens is used to select which table to load. First the user must select a project they wish to load and then a screen within that project. Screens that have no keys and therefore no backing table will not show up in the screen list.

## QC Results Data Chart

QC Results features a data charting module which allows users to visualize their data in a number of different fashions. To create a printable chart, the columns that make up the X-axis and one more series must be chosen. This is achieved by selecting columns of data that need to be charted. A user can select a column by clicking on the column header. The first column selected makes up the X-axis, each column selected after that makes up a series.



ArrivalTime Entry	Refu	Refriger	RefrigeratorOpc	RefrigeratorOpc	TempAtArrival Entry	TempAtArrival Test	AdjustedArrivalTemperature (I)	AdjustedArrivalTemperature (I)
01/21/2011 18:41:38	No		Refused Shipping		11.00	11.00	10.00	10.00
01/25/2011 10:23:47	Yes		Adjusted		-25.00	-25.00	-30.00	-30.00
01/25/2011 08:59:11	Yes		Adjusted		-20.00	-20.00	-12.00	-12.00
01/25/2011 10:22:54	Yes		Adjusted		9.00	9.00	3.00	3.00
01/25/2011 09:01:30	Yes		Refused Shipping		-23.00	-23.00	-30.00	-30.00
01/25/2011 09:02:52	Yes		Adjusted		-21.00	-21.00	-10.00	-10.00
01/25/2011 10:17:18	Yes		Adjusted		-10.00	-10.00	-9.50	-9.50
01/25/2011 10:15:32			Adjusted		-19.00	-19.00	-10.00	-10.00

Figure 13 - QC Table Maintenance with the ArrivalTime, TempAtArrival and AdjustedArrivalTemperature columns selected.

The filter feature of the grid can be used to filter down a given set of results if desired. Once the data columns are selected, the user simply has to choose what type of chart, whether to render it in 2-D or 3-D, and then clicks the 'Create Chart' button.

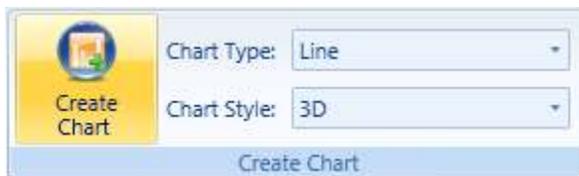


Figure 14 - Create Chart ribbon tool.

A new chart is then rendered and displayed in a new window (see below). The chart can be printed by clicking the print button in the bottom right hand corner of the window.

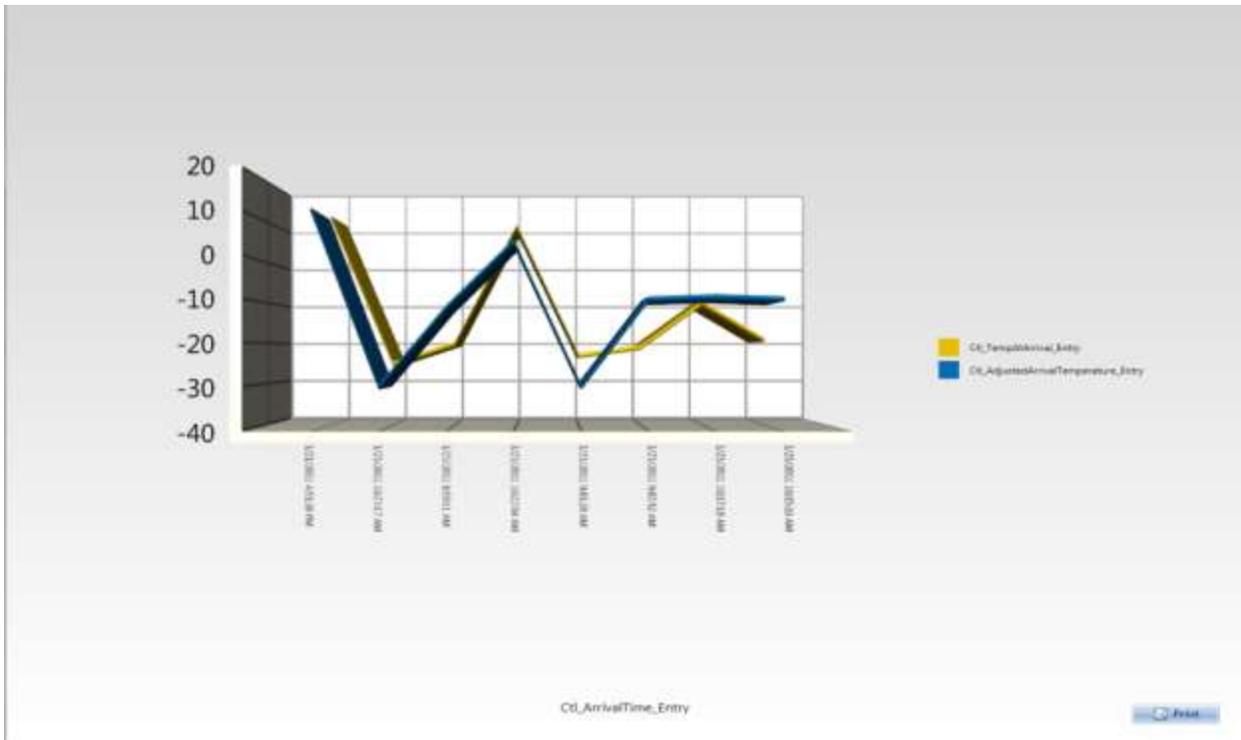


Figure 15 - 3D Line Chart.

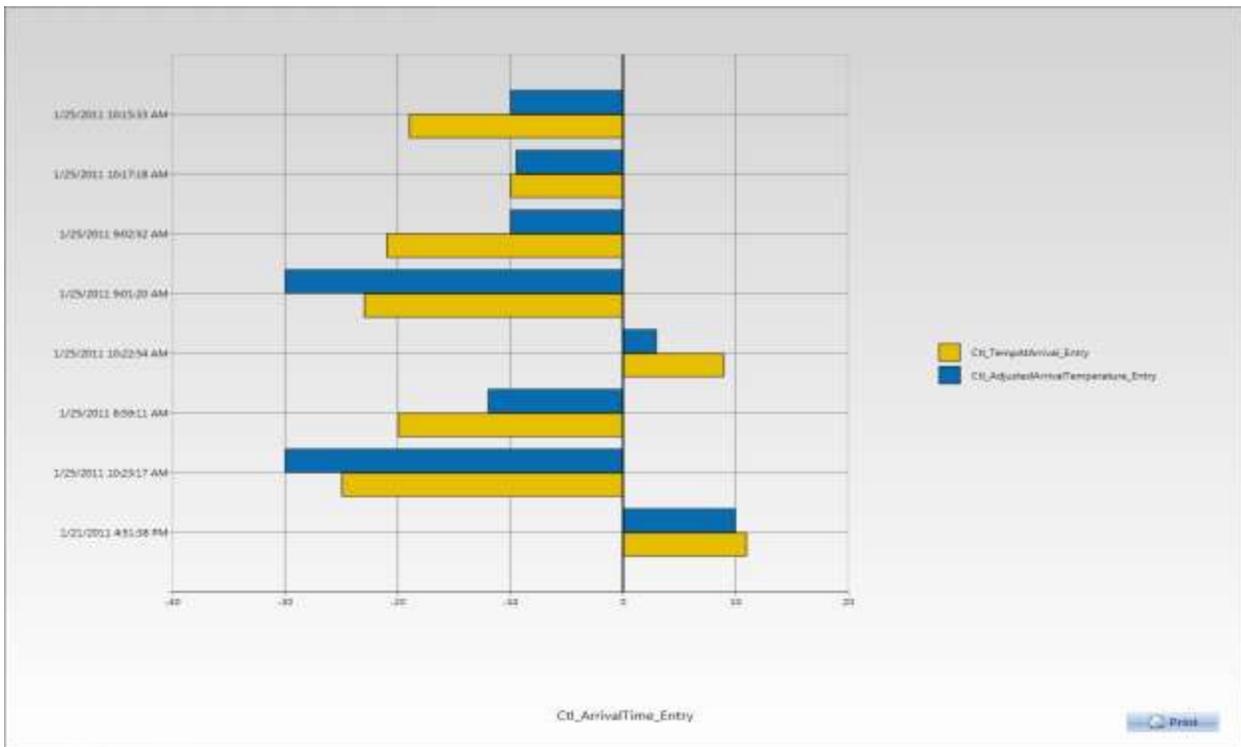


Figure 16 - 2D Bar Chart.

## QC Results Forms Designer



The QC Forms Designer is the tool used to build and maintain QC Results screens. Screen schema data is stored as XML inside the qcScreenSchema database table. Each revision of a screen becomes a separate row in the table. Only one row (revision) per screen can be marked as 'Active' at a time, which is the record that is live and downloaded to QC Results clients at log-in time. Using this approach it is easy to either deploy or rollback screen revisions inside a live environment.

Each time a user saves a new screen revision, a dialog window collects a comment meant to list and detail any changes that were implemented. This can be useful for auditing screen change information later on. In addition to this feature, the tool also tracks and logs changes that were made through internal programming hooks from within the designer itself. These changes are saved to the qcScreenSchemaAudit table, which can be viewed using the QC Schema Revision Manager screen within Whistle Studio.



Figure 17 - QC Forms Designer, with the Controls tab (left) and Code tab (right) visible.

## Using the Tool

The QC Forms Designer tool is designed to be intuitive and for the most part simple to learn and use. It requires only a basic understanding of any C-style programming language. Naturally, in order to build

the more intricate screens it will require more complex use of scripting code. However, the initial set of forms (which can be used as examples) exist and additional support is provided by Coolearth. Anyone who has used Microsoft Visual Studio should at least find the user interface familiar.

### Adding Controls and CoolScript Code Editors

Controls can be added by dragging items from the Control tab (pictured above on the left) onto the active form. CoolScript code event handlers can be added and edited using the Code tab (pictured above on the right). The CoolScript code editor panes (pictured above on the bottom) feature rich text editor controls complete with syntax highlighting and error checking.

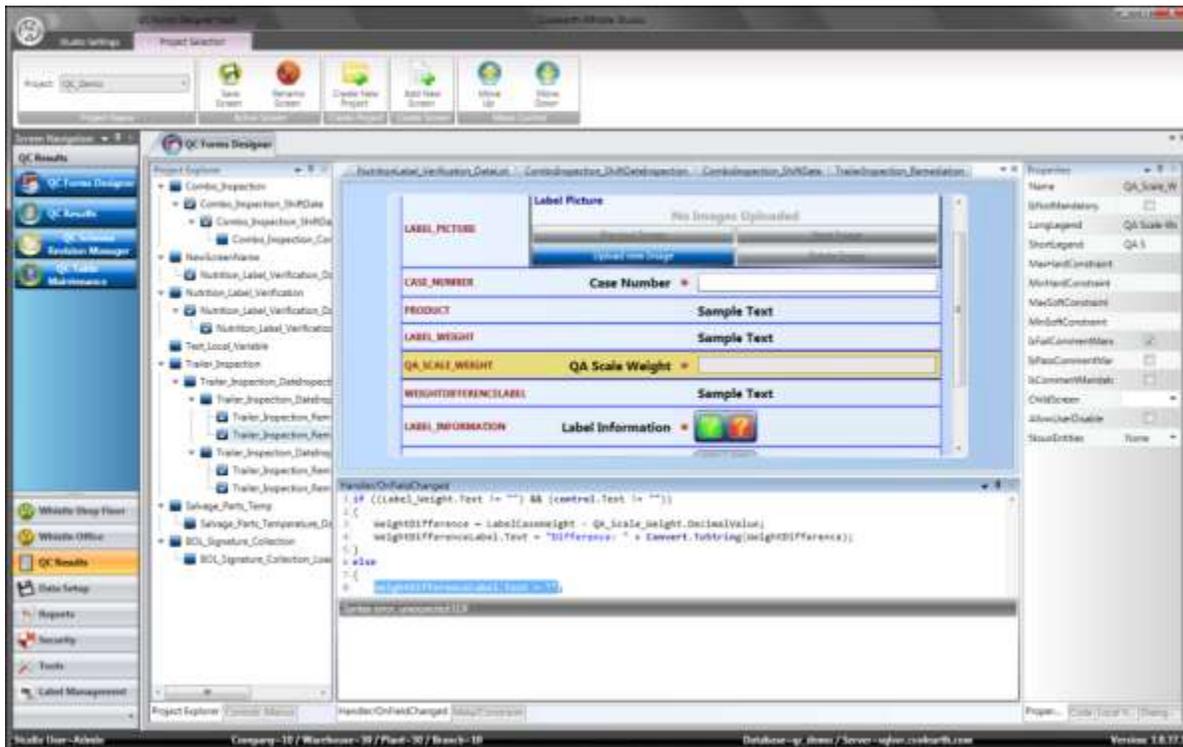


Figure 18 - QC Forms Designer, with the Project Explorer tab (left) and Properties tab (right) visible.

### Project Explorer

A list of the screens and their relationship to their child screens are displayed on a different tab called the Project Explorer. By clicking the check box next to the screen name inside the Project Explorer, screens are either opened for editing or closed. Screens can also be deleted or set to non-active from here.

### Properties Grid

Both controls and screens can have attached attributes that make them function in a certain way, these are known as Control Properties and Screen Properties. Each type of control might have some properties that are specific to how that control works but also some other properties that are common

to all controls (such as, the IsNotMandatory property). The control/screen that is currently selected inside the Designer has its' properties and code elements displayed inside the Properties tab pane (pictured on the right from the screenshot above). In this example, the QA\_Scale\_Weight field is the currently active element.

### Code Generation Wizards

The QC Forms Designer features several CoolScript code generation wizards that can assist the screen programmer in performing many of the common tasks presented while building a screen. For example, a code generation wizard is launched when the 'Add new screen' button is pressed.



Figure 19 - Add New Screen Wizard keys selection screen.

The purpose of the code generation wizard is to collect answers from the person designing the screen. In this case a new screen is being created so the screen name, keys that make up the backing database table and security permissions necessary will all need to be collected. Once all of the answers have been filled out, the tool will automatically generate the CoolScript and/or screen schema information necessary. As new features are added to the system, new code generation wizards will also be added to support those features.

### Saving Changes



All changes that have been made since the screen was loaded can be persisted to the database by clicking the 'Save Screen' button along the ribbon toolbar. After the save change button is clicked a dialog box appears to collect the change-log comments. The new screen revision is then written to the qcScreenSchema table as a new record.

## Controls Glossary

The next section will go into what controls are available inside the QC Results system. These are the building blocks of each screen and get re-used quite frequently.

### Text Editor

The text editor control (also known as a textbox) is possibly the most commonly used control in the system. It supports three different modes of data entry:

Text – The user may enter plain text, this mode does not support constraints.

A rectangular text input field with a light blue border and a yellow inner border. The text "Sample Text.." is entered in a standard sans-serif font.

Integer – Entry is constrained to integer values. This mode supports constraints.

A rectangular text input field with a light blue border and a yellow inner border. The number "10250" is entered in a standard sans-serif font.

Decimal – Entry is constrained to decimal values. This mode also supports constraints.

A rectangular text input field with a light blue border and a yellow inner border. The number "28.53" is entered in a standard sans-serif font.

If the user types in a character that does not parse into a valid numeric value while in Decimal or Integer mode, the keystroke is ignored. Text entered is not validated by any attached CoolScript until the user either leaves the field or hits the Enter key. Negative numeric values are supported.

### Pass/Fail Control



The Pass/Fail control is typically used to collect a Boolean value on whether the item being recorded meets the standards or not. Pictured above is the Pass/Fail control in its default state. The user either clicks the green button on the right for a value of pass (true) or the red button the left for fail (false). The screen designer can choose to embed CoolScript code in either the pass or fail events for this control. For instance, if the user clicks fail, the control can be scripted to launch a linked remediation form (see the *Linked Remediation Forms* section). Once the user has chosen a value of pass or fail and clicked the appropriate button, a comment box will appear.

Two rectangular comment boxes side-by-side. The left box has a green checkmark icon on the left and the text "Sample Pass Comment..". The right box has a red X icon on the left and the text "Sample Fail Comment..". Both boxes have a light blue border and a yellow inner border.

The comment text box provides a means to attach additional information to the control's value. Additionally, the screen designer may opt to make the comment field mandatory for the control to be filled in.

### Quad-State Pass/Fail Control



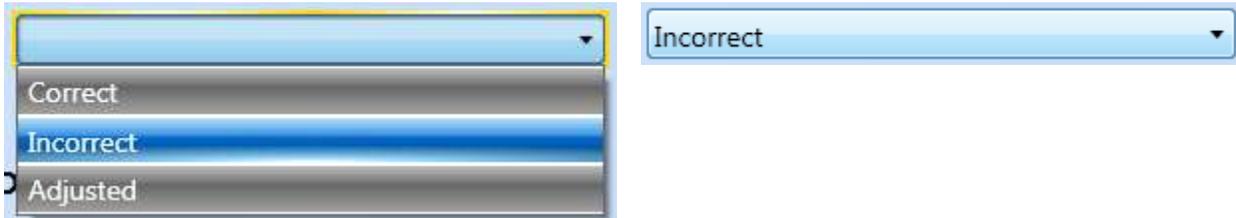
As implied by name, the Quad-State Pass/Fail control is very similar to the regular Pass/Fail control except that there are multiple levels of failure. Instead of a simple Boolean, this control uses an integer to represent its value. The user either clicks the green button on the left for pass, or one of the other buttons for failure. The yellow button represents the lowest level of severity increasing from left to right up to the red button representing the greatest severity. Other than that, this control works exactly like the Pass/Fail control described in the section above.

### Check Box Control



The Check Box control is a very simple control that probably does not need much explanation. One thing to note, though, about the check box control is that it is never mandatory by the very nature of how it works.

### Combo Box



The Combo Box is similar to the Text Editor in that it uses a text value. The user is presented with a predefined list of values and must select one for the control to be considered filled in.

### Date Control



The Date control can be configured to collect either dates or times. Its value is stored as a datetime in the backing SQL Server database tables. There are a couple of things to note about this control. The first is the small button that appears to the right of the date editor.



When clicked, this button will set the date editor's value to the current date/time. When set up to collect dates, rather than typing in the value the user can opt to select a date from the drop down calendar (pictured below).



## Form Link Control



The Form Link control is used to break large screens up into smaller pieces. This can be beneficial in several ways. Having smaller more manageable forms is better from a usability standpoint and requires less scrolling. It also helps conserve resources when running on the hand-held platform where processor speed and memory usage is more limited. Loading up a form with 200 controls makes the screen take longer to load and refresh. Breaking up large forms can also give structure to the form's workflow. For instance, an inspector can be assigned to filling out one section of the form while another inspector is assigned to filling out a different section, therefore making it easy to divide up the work.

Using the Form Link control is simple, it is just a button that when clicked will launch a child screen. The text that appears on the button is customizable through CoolScript. The default text and icon displayed indicate the status of the linked child form. In the screenshot above, there is no form created yet. When the user clicks the button they are navigated to the newly created or already existing linked form. From that point on, the Form Link button will be displayed in one of the following three states:



-- Incomplete, the linked form is in the partial state and still has mandatory fields that need to be filled in.



-- Complete, the linked form is in the complete state, all mandatory fields are filled in but it has not been locked yet.



-- Locked, the linked form has been completed and locked. The fields are contained are read-only.

## Tally Control



The Tally control is for capturing data in real life scenarios that involve counting something. To illustrate, consider the following screen pictured below:



Figure 20 - Tally controls being used to count out-of-spec pieces.

In this example, the numbers of out-of-spec pieces on a pallet are being tallied up. If the number of out-of-spec pieces becomes too high (>3) the entire pallet becomes out-of-spec and corrective action is required. The value of a tally control is always a whole number – it can never be less than 0. Other than that the Tally control is self explanatory, its' value is incremented by clicking the up arrow and decremented by clicking the down arrow. The user also has the option of typing the value in using the numeric edit control.

## Drill-Down Grid

Employee	Flag	AddTime	ChgTime	ChgBy
Alex	Locked	10/27/10 15:31	10/27/10 15:31	rh
Bill	Partial	10/27/10 15:31	10/27/10 15:31	rh
Jennifer	Complete	10/27/10 15:32	10/27/10 15:32	rh
Sam	Partial	10/27/10 15:31	10/27/10 15:31	rh
Sue	Complete	10/27/10 15:31	10/27/10 15:31	rh
Tom	New	10/27/10 15:30	10/27/10 15:30	rh

Figure 21 - Employee Entry Drill Down Grid.

The Drill-Down Grid control is one of the most important controls in the QC Results system because it handles most of the navigation between screens. It represents a one- to-many relationship among parent screens and their children. It differs from the Form Link control in that the Form Link control uses a one-to-one relationship to its child screen. Also, the Drill-Down Grid is used for collecting one or more additional keys that are passed down to the child screen. For instance, in the example above we are collecting employee names as an additional key to be passed down to the next screen.

The user can interact with the Drill-Down Grid in two ways, by either adding a new row (child screen) or viewing/editing an existing row.

### Adding a New Row

By adding a new row, a new instance of the appropriate child screen is created with the keys that are passed down to it. There are two modes for the new row feature, one that requires user input and one that generates a new key value automatically. Which mode it uses depends on the data being collected and how the Screen Designer decides to set it up. In our example, we are collecting employee names so this is not something that can be generated automatically. When the user clicks the 'Add Key' button they are presented with the following dialog:

Employee	Flag	AddTime	ChgTime	ChgBy
Alex	Locked	10/27/10 15:31	10/27/10	
Bill	Partial	10/27/10 15:31	10/27/10	
Jennifer	Complete	10/27/10 15:32	10/27/10	
Sam	Partial	10/27/10 15:31	10/27/10	
Sue	Complete	10/27/10 15:31	10/27/10 15:31	rh
Tom	New	10/27/10 15:30	10/27/10 15:30	rh

Figure 22 - Drill Down Grid with Key Adder dialog displayed.

Once the user fills in the text box and clicks the 'OK' button, they are navigated to a new child screen with the entered value passed down (in this case the employee name: Bob).



Figure 23 - Employee Hygiene screen with Employee name passed down from the parent Drill Down Grid.

The add key dialog can be configured with one of several different types of editor controls such as date editors, numeric editors or a combo box depending on the type of data being passed down.

Notice that the Drill-Down Grid supports the mandatory feature. By default, a mandatory Drill-Down Grid is not considered filled in until it has at least one child row that is in the 'Locked' state. If it has more than one row then each additional row must also have a status of 'Locked'. This is how the QC Results system rolls up the completeness of the child screens into their parents. So if a screen has one or more child screens, then they all must be completed and locked before the parent screen can be considered completed and ready to be locked.

### Viewing an Existing Row

When a user clicks on the 'View Key' button, they are navigated to the existing child screen that corresponds to the currently selected row in the grid. If the child screen has not been locked, they are free to make changes and/or to complete the rest of it.

## Picture Control



Figure 24 - Empty Picture control.

The Picture control is very similar to the control used in the Attach Image Metadata feature. The major difference being that it is embedded as an actual screen control rather than an image that is optionally attached to a field. The mandatory feature is supported, if a Picture control is marked as mandatory then the user must attach one or more pictures for it to be considered filled in. The screenshot above was taken from a screen that is used to collect images of product labels. Here is what the Picture control looks like after a sample image has been collected:



Figure 25 - Picture control with an acquired image.

On the hand-held platform, pictures are taken with the built in camera on the device. For the desktop platform, pictures are uploaded from the user's hard drive.